Freeform Search

Database: US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Term:

Display: 10 **Documents in Display Format:** - **Starting with Number** 1

Generate: ○ **Hit List** [✓ ● **Hit Count**] ○ **Side by Side** ○ **Image**

Search Clear Interrupt

Search History

DATE: Wednesday, December 08, 2004 Printable Copy Create Case

Set Name side by side	**Query**	**Hit Count**	**Set Name** result set
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR		
L13	L12 and (cells or nodes)	55	L13
L12	(hierarch$ or hierarchy or hierarical) near2 (backup or back with up) and retriev$	95	L12
L11	711.clas.	23564	L11
L10	711/162	1673	L10
L9	711/117	940	L9
L8	711/113	1691	L8
L7	711/112	2382	L7
L6	711.clas.	23564	L6
L5	707.clas.	23816	L5
L4	707/205	1809	L4
L3	707/204	2041	L3
L2	707/10	9572	L2
L1	707/1	7243	L1

END OF SEARCH HISTORY

☐ Generate Collection Print

L13: Entry 42 of 55 File: USPT May 28, 2002

US-PAT-NO: 6397308
DOCUMENT-IDENTIFIER: US 6397308 B1

TITLE: Apparatus and method for differential backup and restoration of data in a computer storage system

DATE-ISSUED: May 28, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Ofek; Yuval	Framingham	MA		
Cakeljic; Zoran	Newton	MA		
Gagne; Mathieu	Boston	MA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
EMC Corporation	Hopkinton	MA			02

APPL-NO: 09/ 224897 [PALM]
DATE FILED: December 31, 1998

INT-CL: [07] G06 F 12/00

US-CL-ISSUED: 711/162; 711/112, 711/113, 711/202, 707/204
US-CL-CURRENT: 711/162; 707/204, 711/112, 711/113, 711/202

FIELD-OF-SEARCH: 707/204, 707/205, 711/112, 711/113, 711/162, 711/202

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5535381	July 1996	Kopper et al.	
☐	5555371	September 1996	Duyanovich et al.	395/182.11
☐	5649196	July 1997	Woodhill et al.	
☐	5673381	September 1997	Huai et al.	
☐	5720026	February 1998	Uemura et al.	395/182.04
☐	5751997	May 1998	Kullick et al.	

☐	5778395	July 1998	Whiting et al.
☐	5813017	September 1998	Morris et al.
☐	5852713	December 1998	Shannon et al.
☐	5873103	February 1999	Trede et al.
☐	5925119	July 1999	Maroney
☐	5926649	July 1999	Ma et al.
☐	5950015	September 1999	Korst et al.
☐	6023710	February 2000	Steiner et al.
☐	6047294	April 2000	Deshayes et al.
☐	6052341	April 2000	Bingham et al.
☐	6061822	May 2000	Meyer
☐	6141773	October 2000	St. Pierre et al.

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
0910019	April 1999	EP	

OTHER PUBLICATIONS

International Search Report of International Application No. PCT/US 99/ 29499 mailed May 15, 2000.
U.S. application No. 09/224,637, Ofek et al., filed Dec. 31, 1998.
U.S. application No. 09/224,638, Ofek et al., filed Dec. 31, 1998.
U.S. application No. 09/224,896, Cakeljic et al., filed Dec. 31, 1998.
U.S. application No. 09/223,896, Ofek et al., filed Dec. 31, 1998.
U.S. application No. 09/223,897, Ofek et al., filed Dec. 31, 1998.

ART-UNIT: 2186

PRIMARY-EXAMINER: Kim; Matthew

ASSISTANT-EXAMINER: Vital; Pierre M.

ATTY-AGENT-FIRM: Wolf, Greenfield & Sacks, P.C.

ABSTRACT:

Method and apparatus for generating partial backups of logical objects in a computer storage system are disclosed. Changed data blocks are identified and stored as differential abstract block sets. The differential abstract block set may include data blocks in any order and metadata identifying the relative position of the data block in the logical object. The invention includes methods for formatting updated backups using the differential backups.

24 Claims, 37 Drawing figures

Previous Doc Next Doc Go to Doc#

L13: Entry 42 of 55 File: USPT May 28, 2002

DOCUMENT-IDENTIFIER: US 6397308 B1
TITLE: Apparatus and method for differential backup and restoration of data in a computer storage system

Brief Summary Text (9):
The application level 10 interfaces with the file system level 12. The file system level is concerned with how files are stored on disks and how to make everything work efficiently and reliably. Thus, the file system level may be responsible for storing directory structure, and for breaking up files into constituent data blocks for storage onto a physical storage system. For example, in most implementations of Unix, each file has an associated I-node. This node may contain accounting and protection information and, additionally, a set of pointers to data blocks.

Brief Summary Text (60):
According to one embodiment of the present invention, a computer system is disclosed. According to this embodiment, the computer system includes a host domain that has at least one host computer. The computer system also includes a storage domain, coupled to the host domain, that comprises a plurality of primary storage devices, a secondary storage device and a switched network coupled to the primary storage nodes and to the secondary storage node.

Brief Summary Text (62):
According to another embodiment of the present invention, a method of transferring, data from a primary storage node to a secondary storage node is disclosed. According to this embodiment, a connection is automatically established from one of the primary storage elements to a secondary storage element, for transferring data to the secondary storage element. Data is transferred from the primary storage element directly to the secondary storage element over the first connection.

Drawing Description Text (32):
FIG. 27 illustrates one example of a system for backing up data on a primary storage node, using a secondary storage node, according to one embodiment of the present invention.

Drawing Description Text (35):
FIG. 30 illustrates one embodiment of a system and data flow within a system for sending copy of backup information from a primary storage node.

Drawing Description Text (36):
FIG. 31 illustrates one embodiment of a method for sending data from a primary storage node.

Detailed Description Text (5):
The host computers 80 in the enterprise host domain 88 may be connected over a network. This network may include switching nodes 81, although any other form of network may be used.

Detailed Description Text (6):
In the embodiment of FIG. 8, the host computers 80 are coupled to the enterprise storage 89 through a network or directly to primary storage nodes 82. A primary

storage node is a memory device that can store significant amount of data for use by the host 80. For example, a Symmetrix system, such as the one described above with respect to FIG. 7, may be used as a primary storage node, although this is not intended as limiting.

Detailed Description Text (7):
In the embodiment of FIG. 8, each host computer is coupled to a subset of primary storage nodes 82, for use as a main memory for that host computer. For example, host computer 80a is coupled directly to primary storage node 82a. The host computer 80a may rely on primary storage node 82a for most of its memory intensive functions, such as for accessing a very large database.

Detailed Description Text (8):
The primary storage nodes 82 may also be coupled together through a network. In the example of FIG. 8, the network includes link 85 and switch network 84. The switch network 84 may, for example, be a fiber channel network. The link 85 may be an RDF link over an ESCON line.

Detailed Description Text (9):
The network between primary storage nodes may serve two purposes. The network may permit transfer of data between primary storage nodes. For example, a database being manipulated by host 80a, and stored in primary storage node 82a, may be transmitted to primary storage node 82b for use by host 80b. By transmitting the database across the enterprise storage network (using link 85 or switching network 84), the computational resources of the host 80a, 80b, and the available bandwidth in the enterprise host domain network, can be preserved.

Detailed Description Text (10):
The enterprise storage network 89 may also include a secondary storage node 87. The secondary storage node may be used for backup functions, hierarchical storage management, virtual disks and other functions. Thus, the secondary storage node 87 may be coupled to a tape storage unit 83. The secondary storage node 87 would coordinate sophisticated transfer of data from the primary storage nodes 82 to the tapes stored in a tape storage unit 83. (Other embodiments may use additional or alternative media for secondary storage.)

Detailed Description Text (12):
In the embodiment of FIG. 9, each host computer 90a-90e is connected to a primary storage node 92a-92c. In this embodiment, each primary storage node 92a-92c is an iterative cached disk array, such as a Symmetrix memory system such as the one described above with respect to FIG. 7, although this is not intended to be limiting. Thus, for example, host computer 90a interfaces primarily with storage node 92a. Similarly, host computer 90b uses primary storage node 92a as a primary source of its data.

Detailed Description Text (13):
In the embodiment of FIG. 9, the host computer 90a is connected to the primary storage node 92a over a high speed fiber channel 91a. The host 90b, however, is connected to the primary storage node 92a over a standard SCSI connection. Each of the hosts 90a and 90b are coupled to the same primary storage node 92a. Other mechanisms could be used to connect the host computers 90a-90e to the primary storage nodes 92a-92c. For example, a complete switched network could be employed, for any of the host computers to access any of the primary storage nodes 92a-92c.

Detailed Description Text (14):
Each of the primary storage nodes 92a-92c may also be coupled together using a network. In the example of FIG. 9, the only link among the primary storage nodes is an ESCON remote data facility (ESCON "RDF") link 93g. Such a link may be used for transferring of data or maintaining a mirror of data either on-line or as a periodically updated mirror. Such a link may be implemented as described in U.S.

Pat. No. 5,544,347 (Yanai), which is incorporated herein by reference in its entirety. Each of the primary storage nodes 92a-92c may be coupled together using any other mechanism. For example, an RDF link could be used to fully connect each of the primary storage nodes 92a-92c. In the alternative, a switch network could be used, assuming that the network is of sufficiently high speed to support the data operations among the primary storage nodes 92a-92c.

Detailed Description Text (15):
The storage network 98 in the embodiment of FIG. 9 further includes a secondary storage node 94. The secondary storage node is used for backup (and other) functions, for example by storing and restoring information to and from a tape library 95.

Detailed Description Text (16):
In the embodiment of FIG. 9, each of the primary storage nodes is connected or connectable (by a network) to the secondary storage node 94. In this example, primary storage nodes 92b and 92c are coupled to secondary storage node 94 each using an RDF link (93c and 93d respectively) which may be implemented as described above.

Detailed Description Text (17):
The primary storage node 92a is connected (together with other primary storage nodes, not shown) to the secondary storage node 94 over a switched network, which will permit each of the systems to access the secondary storage node 94.

Detailed Description Text (18):
Using an RDF (or other) link that permits high speed transfer of data over long distances, the primary storage nodes 92a-92c and the secondary storage device 94 may be physically located at great distances apart.

Detailed Description Text (22):
In the storage-centric model, however, the storage component of the computer system is elevated to a status of equal importance. In such a model, the storage components of the system are capable interacting with each other with less involvement from the host domain. For example, it may be desirable to permit mirroring across one or more primary storage nodes. Similarly, data objects may need to be copied from one primary storage node to another primary storage node. Where additional levels of backup are desirable, the primary storage nodes may also transfer data to a secondary storage node for backup purposes. The primary storage nodes may, correspondingly receive data from the secondary storage nodes for restore. In a storage centric model, some or all of the resource intensive functions in such a system can be moved out of the host domain. Certain embodiments following this model can preserve host domain resources, increase scalability of memory (by adding to the storage domain without as much concern about affect on host domain resources) and reduce dependence on the particular platforms of the hosts in the host domain.

Detailed Description Text (24):
For example, for a copy, the physical elements that are to be copied are identified at step 100. In addition, the location of where the elements are to be copied to are identified. For a copy between primary storage nodes, this may involve identifying the copy from locations and the copied to locations. For a backup, this involves identifying the copy from locations and may be as simple as determining what tape or other backup storage element will receive the backup data.

Detailed Description Text (25):
For a copy between primary storage nodes, the physical elements are transferred from the identified copy from locations to the identified copy to locations. For a backup, the physical elements are copied to tapes. (Although reference is made to tapes as secondary storage, this is not intended to be limiting. Any other storage

media may be used).

Detailed Description Text (26):
The step 100 can, however, be extremely complicated. In many cases, it is not desirable to copy the entire contents of a primary storage node. Rather, only a subset of the physical elements in the primary storage node may need to be copied. As one example, consider backing up a database stored in primary storage node 92a of FIG. 9. This database may occupy only a small portion of the total data stored in the primary storage device 92a--in fact, there may be an extremely large segment of data accessible primarily by the host computer 90b which host 90a may not even be capable of reading (because it is a different platform than the host computer 90a).

Detailed Description Text (27):
In short, it may be desirable to backup a logical object stored within a primary storage node. In this case, the step 100 requires mapping the logical object onto the physical elements in the primary storage node 92a in order to identify the physical elements that need to be copied from 92a. As described above with reference to FIG. 2C, these physical elements may be located in disparate locations within the primary storage device.

Detailed Description Text (28):
The step 102 may similarly be complicated. Even after all of the physical elements in the primary storage device have been identified, simply transferring the physical elements is insufficient. The relationship between the physical elements may need to be preserved for the copied or backed-up logical object to be read by the host computer coupled to the receiving primary storage node. One mechanism for use of mapping a logical object to physical elements and preserving the logical relationship between those physical elements is discussed below. This is not intended as limiting with respect to other aspects of the present invention.

Detailed Description Text (29):
In any event, under a storage-centric model of computer storage, it may be desirable to permit as much of the data transfer process (e.g., the one shown in FIG. 10) to be performed within the storage network--and without requiring resources from the host domain. Accordingly, the primary storage nodes and the secondary storage nodes in the network may include sufficient intelligence to handle aspects of the data transfer process. For example, the primary storage nodes may be capable, at a minimum, of managing the transfer of identified physical elements in a logical object even when those physical elements are stored in disparate locations within the primary storage device. In a storage centric model of a computer system, it may be desirable to move some (or as much as possible, in some cases) of the data transfer functions to be performed using resources among primary and secondary storage nodes within the storage domain.

Detailed Description Text (30):
The computer system may include a storage management application ("SMAPP") for managing manipulation of storage within the storage domain. The SMAPP can be implemented using software on the host computers, primary storage nodes, a separate storage controller or in some combination of these, as described below with reference to FIGS. 11A and B, below.

Detailed Description Text (38):
In this example, the primary storage element includes an SMAPP interface 116a. Similarly, the secondary storage element 112 includes an SMAPP interface 116b. The copying of a logical object from the primary storage element 111 to the secondary storage element 112 in the embodiment shown in FIG. 11A may proceed as follows. First, a "virtual circuit" or "connection" is set up between the primary storage element 111 and the secondary storage element 112. This may be a virtual circuit established through a network coupling the primary storage element to the secondary

storage element 112 (including a single RDF link between the primary storage element 111 and the secondary storage 112, for example). In addition to establishing a physical connection between the nodes, the virtual circuit identifies a session for copying a series of data (comprising, e.g., the logical object) over the identified connection.

Detailed Description Text (57):
As described above, there are at least two different ways of passing data blocks of a logical object to a storage element--transferring the blocks in order as a logical object (as is done over a network between host computers) and a pure physical copy (which may not preserve the logical relationship among the data). Each of these possibilities has advantages and disadvantages. For example, copying each data block of a logical object in order preserves the relationship between data blocks. On the other hand, copying the blocks in order may result in delays as the storage elements sequentially retrieve the data blocks or sort the data blocks for writing, as a part of the copy process. On the other hand, pure copying of physical elements can be unnecessarily slow if unused physical elements are copied. In addition, the logical relationship between the data blocks that are copied may be lost.

Detailed Description Text (80):
FIG. 15 illustrates one embodiment of a method for restoring an abstract block set to a memory system, such as the primary storage node described above.

Detailed Description Text (81):
At a step 150, the metadata for the abstract block set is retrieved. This may be in the form of a map for the abstract block set such as those illustrated at 134 of FIG. 13 or may be a set of labels associated with the individual data blocks stored in the abstract block set, such as in table 133 of FIG. 13.

Detailed Description Text (86):
If not, at a step 156, the next physical backup segment is retrieved. At a step 157, the location and the newly allocated memory for receiving the logical object is determined. This can be done by examining the re-mapping table created at step 153. In addition, the retrieval of segments done at step 156 need not be in any specific order. The re-mapping table permits restoration of the entire logical object even when the data blocks are provided in a random order.

Detailed Description Text (179):
Primary to Secondary Storage Node Transfers, Example of One Secondary Storage Node.

Detailed Description Text (180):
As described above with respect to FIGS. 11A and 11B, one aspect of storage systems involves transfer of data from primary storage elements or nodes to secondary storage elements or nodes.

Detailed Description Text (181):
FIG. 27 illustrates one example of a particularly advantageous mechanism for transferring data from a primary storage node to a secondary storage node for storage on tape. This example embodiment and the components of FIG. 27 are useful both in the context of the other inventions described above (although not limiting with respect to those inventions), as well as useful for systems implemented independent of those inventions.

Detailed Description Text (182):
FIG. 27 includes a primary storage node 270. This may be, for example, a Symmetrix storage system as described above. In such a system, a host adapter 270a may be provided for communication with a host. Disk adapters may provide an interface with the disks. A remote adapter 270c may handle communications with remote devices,

whether through a SCSI link, an ESCON link, a fiber channel, a switched network, or some other communication channel. In addition, a cache 270b may be provided for caching received and transmitted data.

Detailed Description Text (183):
FIG. 27 also illustrates a secondary storage node 271. In this embodiment, the secondary storage nodes has a plurality of data moving elements 271a, 271b, 271e and 271f. In this embodiment, the data moving elements are arranged in pairs--a front end and back end pair. For example, data mover 271 a may be a front end data mover--primarily responsible for receiving data from a primary storage node. The front end data mover 271 a may be paired with a back end data mover 271e. The back end data mover is responsible for moving data from the secondary storage node to the backup media.

Detailed Description Text (186):
Returning to the secondary storage node 271, the secondary storage node may include an internal storage device 271c for buffering data received from the front end data mover (e.g., 271a), before being written to tape by the back end data mover (e.g., 271e) during a backup (or, conversely, for buffering data during a restore by placing the data in the internal memory 271c (by a backbend data mover 271e) and forwarding the data to a primary storage node (by front end data mover 271a).

Detailed Description Text (187):
The data movers 271a, 271b, 271e and 271f may be Intel based personal computers, running software permitting the data movers to transfer data from the primary storage node to the tape library unit during backup, and vice versa during a restore.

Detailed Description Text (189):
The front end data mover (e.g., 271a) may be connected to the primary storage node 270 using any of a variety of connections. For example, in the example of FIG. 27, two ESCON cables are used to connect each front end data mover to the ports of a remote adapter of a single primary storage node (e.g., a Symmetrix storage device).

Detailed Description Text (195):
As described above, the primary storage node 270 may be used as the interface between host connectors (e.g., host computers connected to host adapter 270a) and secondary storage node, 271. In these embodiments, and where the storage management application resides primarily on the host computer, the primary storage node 270 may be used to pass commands from the host computer to the secondary storage node 271. Such commands may include instructions directed to mounting and dismounting tapes, reading and writing tape headers and trailers and other commands.

Detailed Description Text (196):
The primary storage node 270 may simply pass appropriate commands to the secondary storage node 271. In the alternative, the primary storage node 270 may perform some functions based on those commands, such as format checking.

Detailed Description Text (197):
As described above, the backup restore process can be performed by establishing a virtual channel between a primary storage node 270 and the tape library 272, through the secondary storage node 271. As described above, this may involve formulating a connection through a network between primary storage node 270 and secondary storage node 271. This may also involve establishing a connection with a tape drive 272a and applicable tapes 272g.

Detailed Description Text (198):
FIG. 28 illustrates one example of a state diagram for a secondary storage node, such as node 271, for establishing and maintaining a virtual channel. At state 280,

a backup control stream session (or virtual channel) is requested by the storage management application (e.g., on the host computer). Establishment of the virtual channel may involve selecting an appropriate front end and back end data mover pair, e.g., front end data mover 271a and back end data mover 271c.

Detailed Description Text (199):
A function to be performed by the storage management application may require opening a tape. The result would be to place the secondary storage node 271 into state 281--virtual channel beginning of tape. This transition would involve mounting the appropriate tape, using similar techniques to what is known in the art. At the beginning of tape state 281, tape headers and trailers may be read or written, as a part of the tape management process.

Detailed Description Text (200):
When it is time to record information on the tape, the secondary storage node 271 (or at least the applicable data movers within the secondary storage node) enter the virtual channel write state 282. When in this state, the recording part of a backup is performed, such as writing one or more abstract block sets, or portions of an abstract block set, to tape.

Detailed Description Text (201):
If the end of a tape is encountered, the applicable data movers in the secondary storage node 271 enter the virtual channel end of tape state 284. In this state, the applicable catalog information may be read and an appropriate tape trailer written. When the end of the tape is encountered (or end of data), the applicable virtual channel needs to close that tape, returning the data movers and the secondary storage node to the initial state when the channel was formed--state 280.

Detailed Description Text (203):
As discussed above, the storage management application is responsible for issuing the appropriate commands to change the state of the secondary storage node 271. The storage management application may be resident on the host computer, primary storage nodes, separate network storage controller or even on the secondary node 271.

Detailed Description Text (204):
FIG. 29 illustrates a state diagram for the secondary storage node 271 for restoring information from tape. The state diagram begins at state 291, where a request to open a virtual channel has been received. The storage management application handles the opening of tapes, for example by requesting a tape open for the backup channel stream. This results in entering the virtual channel beginning of tape state 292. As before, this can include tape header and trailer reads as well as reading of abstract block set metadata, for systems using abstract block sets.

Detailed Description Text (205):
The actual reading of data can be controlled using a tape read command, causing the secondary storage node 271 to enter into the virtual channel read state 293. At end of tape (or data) or log-out, the secondary node may return to the virtual channel end of tape state 292. The tape may then be closed, returning the secondary storage node 271 to the virtual channel opened state.

Detailed Description Text (206):
If an error is encountered during reading, the node 271 may enter the error state 294, similar to the error state described above with reference to FIG. 28. When an error occurs, the tape may be closed, an error log created, and the system operator notified.

Detailed Description Text (207):

For both backup and restore, the cataloging and identification of tapes can be handled by the storage management application, as is done for other mechanisms for formatting data stored on a storage system. The control station 271g of the secondary storage node 271 assists in identification and mounting and dismounting of the appropriate tapes, using the control station database 271i.

Detailed Description Text (208):
The backup and restore state diagrams of FIGS. 28 and 29 constitute example embodiments of placing the system (e.g., the primary storage node and of the secondary storage node) in an asynchronous transfer stale. In particular, the nodes of the storage domain enter a state where data is transferred independent of control from any host computer or host domain element, even when much of the storage management application process (and software) is being performed on the host computer.

Detailed Description Text (211):
FIG. 30 illustrates one embodiment of an architecture for a primary storage node that facilitates transfer of data to a secondary storage node or to another primary storage node. This embodiment (as well as others) may be used to implement one or more of the above inventions.

Detailed Description Text (212):
FIG. 30 illustrates a primary storage node 300. The primary storage node 300 includes a remote adapter 301, as generally described above with reference to FIG. 7. The primary storage 300 also includes a disk adapter 305, also configured as generally described above with respect to FIG. 7.

Detailed Description Text (213):
Data is stored among a plurality of disks within the primary storage node 300, one of which is shown in FIG. 30--disk 306.

Detailed Description Text (216):
Those physical backup segments (e.g., tracks 308a, 308b and 308e) that were designated as part of a backup process may then be copied to a side file 303 in a cache 302 of the primary storage node 300. Thus, the side file 303 may receive the designated tracks 308a, 308b and 308c for copying to another storage node. The side file, therefore, may contain copies 303a-c of these tracks.

Detailed Description Text (217):
In addition, the disk adapter 305 may post, to a request queue, a request that the physical backup segments that have been copied to the side file 303 be transferred to another node. Thus, requests 304a-c may be posted in the request queue 304, corresponding to those physical backup segments in the side file 303.

Detailed Description Text (218):
The remote adapter 301 may pickup requests from the queue and transfer copies of the applicable track to the receiving storage node, e.g., a secondary storage node.

Detailed Description Text (220):
In an alternative embodiment, the receiving storage node may classify physical backup segments based on the abstract block set to which they belong. For example, the front end data movers described above could receive physical backup segments corresponding to tracks, including a physical address for the track. The front end data move may be aware of the metadata for the abstract block set, which was formulated by the storage management application (which identified all of the physical locations for the applicable logical object being backed up). This would permit the front end data mover to classify the physical backup segment based on its physical address.

Detailed Description Text (221):
Of course, a variety of alternative structures and methods could be employed for transfer through a side file. As just one example, the physical backup segments could be sorted into separate side files for each abstract block set (or other structure) being copied or backed up. In addition, side files may be used to accumulate segments of data for transfer. For example, a side file could be created that includes at least ten megabits of data before transfer through the remote adapter 301 to a secondary, or other, storage node.

Detailed Description Text (223):
In addition, metadata for the applicable logical object may be transferred to the receiving storage node, e.g., the secondary storage node. Thus, if the metadata is of the form shown at 133 of FIG. 13, this metadata may be specified and advance the backup process. This metadata may (or may not) be reformulated during backup for incorporation into the logical backup object, such as reformulation into the form shown at 134 of FIG. 13. In any event, this metadata may be used by the disk adapter 305, remote adapter 301 and/or the receiving storage node to accumulate and organize the applicable physical segments associated with the logical object being copied or backed up.

Detailed Description Text (224):
At a step 311, the protected segments are transferred to a side file in a cache. As this is done, requests for the transfer of the physical backup segments are logged into a request queue. As described above, this may be performed by a disk adapter of the primary storage node. At this point in time, the disk adapter 305 may also reset the applicable protection bit of the protection bits 307 of the disk device 306, allowing future updates of the data.

Detailed Description Text (225):
The segments in the side file can then be transferred to another storage node by the remote adapter 301, such as transfer to a secondary storage node. This may be done be reading requests for transfer from the requests queue 304.

Detailed Description Text (227):
FIG. 32 illustrates one example of data flow in a backup process through a secondary storage node 320. In this embodiment, the data is initially received by front end processor 322. The front end processor may be as generally described above with reference to FIG. 27.

Detailed Description Text (233):
In FIG. 33 the tape has a beginning portion 330 and an ending portion 332. The beginning portion 330 includes the usual tape header 330a, and perhaps a specific tape header for the secondary storage node 330b. After the tape headers 330a, 330b, the tape includes interleaved segments of abstract block sets (including metadata) 338, separated with file marks. For example, the interleaved segments may include a record 331 that includes a series of copies of physical backup segments 331b. A segment header 331a and segment trailer 331c may identify and separate this portion of the abstract block set from other portions of the tape.

Detailed Description Text (236):
Using a database of tapes, the applicable information may be retrieved from a backup tape. Because abstract block sets may include data blocks written in any order, a restore process can efficiently retrieve and write the portions of an abstract block set being restored, in any order. This permits the storage management application to identify each of the tapes that include portions of an abstract block set and to mount (and read all of the applicable portions of) those tapes only once. Of course, the first tape to be mounted may be the tape that includes the metadata records for the abstract block set being restored. For this reason, it may also be preferable to record the metadata at one end of all of the segments of an abstract block set written on the tape holding the metadata--making

the reading of metadata at the beginning process simpler. This permits formation of the appropriate mapping table, described above, for the restoration process to proceed independent of the order in which data blocks are retrieved.

Detailed Description Text (237):
For the reasons described above, the reading and restoring of data blocks within an abstract block set can be done in any order. As a result, where tapes are used and as a component of the secondary storage element, the tapes can be mounted and dismounted in any order for both storing and retrieving data. As a result, where more than one tape drive is present in the secondary storage element, it is shown in the embodiments described above, data blocks can be written during backup and read during restore and parallel using multiple drives.

Detailed Description Text (239):
One example of a parallel restore operation may be described with reference to FIG. 15. As described above, at steps 150-153, the mapping for the restore of the logical object is determined. Where this restore is coming from a tape, the metadata for the abstract block set can be retrieved in advance. As described above, after this has been done, the abstract block sets can be restored in any order. Accordingly, the abstract block sets may also be retrieved in parallel using multiple tape drives for a restore. In this case, the steps 154-158 may be performed in parallel using multiple tapes or other media) for retrieving data blocks of the abstract block set being restored.

Detailed Description Text (240):
In embodiments employing virtual channels, a separate virtual channel may be established for each of the parallel paths for transfer of data. For example, a separate virtual channel may be established for each tape drive. In another embodiment, a single virtual channel may be established, but permitting multiple tape drives to channel data into that virtual channel. This may be particularly advantageous where the speed of reading data from the tape drive is slower than the ability to transfer data from the secondary storage node to a primary storage node. Allowing parallel reading of tape drives permits the speed of the restore to approach the ability of the connections to transfer data and the primary storage element to receive that data.

Detailed Description Text (241):
While many of the above embodiments have been described with respect to backup and restore operations between a primary storage element and a secondary storage element, many aspects of the invention have much broader application. As just one example, an abstract block set can be used for any transfer of data. As another example, the application of a secondary storage node can be greater the simply backup and restore operations. Such storage nodes may also be used for hierarchical storage management applications, operation of virtual disks, and other applications.

Previous Doc Next Doc Go to Doc#

☐ Generate Collection Print

L13: Entry 50 of 55 File: USPT Jul 4, 2000

US-PAT-NO: 6085233
DOCUMENT-IDENTIFIER: US 6085233 A

TITLE: System and method for cellular network computing and communications

DATE-ISSUED: July 4, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Jeffrey; Francis	Malibu	CA		
Benster; Richard W.	Woodside	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Pankosmion, Inc.	Malibu	CA			02

APPL-NO: 08/ 580921 [PALM]
DATE FILED: December 29, 1995

INT-CL: [07] G06 F 13/38, G06 F 15/17

US-CL-ISSUED: 709/216; 709/212, 709/237, 709/248, 709/300
US-CL-CURRENT: 709/216; 709/212, 709/237, 709/248, 719/310

FIELD-OF-SEARCH: 395/800, 395/180, 395/683, 395/702, 395/705, 709/300, 709/303, 709/205, 709/212, 709/213, 709/237, 709/248, 709/216

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	3287649	November 1966	Rosenblatt	328/55
☐	4149240	April 1979	Misunas et al.	395/800
☐	4675829	June 1987	Clemenson	364/513
☐	4864497	September 1989	Lowry et al.	364/300
☐	4930071	May 1990	Tou et al.	364/300
☐	5119470	June 1992	Highland et al.	395/64
☐	5159685	October 1992	Kung	395/575

☐	5193180	March 1993	Hastings	395/575
☐	5404550	April 1995	Horst	395/800
☐	5574933	November 1996	Horst	395/800
☐	5692193	November 1997	Jagannathan et al.	395/676

OTHER PUBLICATIONS

J. Philbin, "An Overview of the STING Operating System," Proceedings of the 4th NEC Software Conf., Oct. 1992.
S. Jagannathan, "Customization of First-Class Tuple-Spaces in a Higher-Order Language," Conf. on Parallel Architectures and Languages, Europe, Springer-Verlag LNCS 506 (1991).
J. Philbin et al., "Efficient Support for Multiple Concurrency Paradigms in Modern Programming Languages," Conf. on Parallel Architectures and Languages, Europe, Springer-Verlag LNCS 506 (1991).
Rogers et al., Supporting Dynamic Data Structures on Distributed Machines, ACM Transactions on Programming Languages and Systems, vol. 17, No. 2, pp. 238-263, Mar. 1995.
G. C. Hill, "Cyber Servants Electronic `Agents` Bring Virtual Shopping A Bit Closer to Reality," The Wall Street Journal, Sep. 27, 1994.

ART-UNIT: 276

PRIMARY-EXAMINER: Rinehart; Mark H.

ATTY-AGENT-FIRM: Wilson Sonsini Goodrich & Rosati

ABSTRACT:

A cell-based system for computation and communication. A cell is a well-defined structure that associates executable code and data into a basic computational module. A processor can traverse a cell and execute the instructions therein. Cells are arranged into a network with each cell linked to other cells by forward branches or other paths. A processor branches to a cell by loading its program counter with the address of the cell or by sending a packet of data across a communications network to activate remote processing in a cell. Cells have a path selection section for evaluating conditional branches and causing branches to be followed. Multiple branches may be followed in parallel from any cell. A tree-like organization is superimposed on the network by distinguishing one branch to each cell as a superbranch incorporating a return path (called "recession path"). A cell also has a convergence section that handles processors and threads of execution that return to that cell by this path (called "recession"). The convergence section implements rules that control which, if any, processors or threads continue to execute upon recession as well as rules for handling any resulting values associated with the processors or threads. The cellular computational module controls highly parallel and distributed processing and supports dynamic self-modification of the system. Information is acquired within the system not only through the acquisition of data and the modification of executable code within the cells, but also through the encoding of information and behavior into the structure of links that connect the cells.

14 Claims, 30 Drawing figures

Previous Doc Next Doc Go to Doc#

L13: Entry 52 of 55 File: USPT Jul 1, 1997

US-PAT-NO: 5644766
DOCUMENT-IDENTIFIER: US 5644766 A

TITLE: System and method for managing a hierarchical storage system through improved data migration

DATE-ISSUED: July 1, 1997

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Coy; Henry Robert	Boca Raton	FL		
Rees; Robert M.	San Jose	CA		
Cabrera; Luis Felipe	San Jose	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY			02

APPL-NO: 08/ 697397 [PALM]
DATE FILED: August 23, 1996

PARENT-CASE:
This application is a continuation of U.S. application Ser. No. 08/198,972, filed Mar. 22, 1994, now abandoned.

INT-CL: [06] G06 F 13/00, G06 F 12/00

US-CL-ISSUED: 395/620; 395/621, 395/444, 395/488, 364/246, 364/283.2, 364/DIG.1
US-CL-CURRENT: 707/204; 707/205, 711/161

FIELD-OF-SEARCH: 395/600, 395/441, 395/488, 395/497.02, 395/848, 395/444, 364/228.1, 364/222.81, 364/246.3, 364/243, 364/282.1, 364/246, 364/283.2

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4183083	January 1980	Chatfield	364/200
☐	4817050	March 1989	Komatsu et al.	364/900

☐	4888681	December 1989	Barnes et al.	364/200
☐	4974156	November 1990	Harding et al.	364/200
☐	4987533	January 1991	Clark et al.	364/200
☐	5018060	May 1991	Gelb et al.	364/200
☐	5133065	July 1992	Cheffetz et al.	395/575
☐	5237682	August 1993	Bendert et al.	395/600
☐	5276860	January 1994	Fortier et al.	395/575
☐	5276867	January 1994	Kenley et al.	395/600
☐	5367698	November 1994	Webber et al.	395/800
☐	5423034	June 1995	Cohen-Levy et al.	395/600

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
1-173236	July 1989	JP	
4-107750	April 1992	JP	
4-165541	June 1992	JP	
3-273275	April 1993	JP	

OTHER PUBLICATIONS

W.D. Roome, "3DFS: A Time-Oriented File Server," proceedings of the Summer USENIX Conference 1991.
R.M. Bryant and P.A. Franaszek, "Method for Allocating Computer Disk Space To A File Of Known Size" IBM TDB vol. 27, No. 10B, Mar. 1985, pp. 6260-6261.

ART-UNIT: 237

PRIMARY-EXAMINER: Black; Thomas G.

ASSISTANT-EXAMINER: Robinson; Greta L.

ATTY-AGENT-FIRM: Klein; Esther E.

ABSTRACT:

A system and method are provided for preserving spacial and temporal locality of sets of related objects when moving the sets within a storage hierarchy via a common server. The appropriate meta data is gathered to track the spacial and temporal locality of the sets of objects being moved within the storage hierarchy and the algorithm uses the meta data to preserve the spacial and temporal locality when moving the objects. A collection of logically clustered data objects is identified. The logical cluster is then moved down through the storage hierarchy together to be stored in less costly storage devices. The logical cluster of data objects is then retrievable more efficiently as a whole when requested.

20 Claims, 7 Drawing figures

Previous Doc Next Doc Go to Doc#